Multiple fundamental frequency estimation based on sparse representations in a structured dictionary ☆,☆☆

Michal Genussov, Israel Cohen *

Department of Electrical Engineering, Technion – Israel Institute of Technology, Haifa 32000, Israel

A R T I C L E   I N F O
Article history:
Available online 11 September 2012
Keywords:
Piano transcription
Music information retrieval
Sparse representations
Multi-pitch estimation

A B S T R A C T
Automatic transcription of polyphonic music is an important task in audio signal processing, which involves identifying the fundamental frequencies (pitches) of several notes played at a time. Its difficulty stems from the fact that harmonics of different notes tend to overlap, especially in western music. This causes a problem in assigning the harmonics to their true fundamental frequencies, and in deducing spectra of several notes from their sum. We present here a multi-pitch estimation algorithm based on sparse representations in a structured dictionary, suitable for the spectra of music signals. In the vectors of this dictionary, most of the elements are forced to be zero except the elements that represent the fundamental frequencies and their harmonics. Thanks to the structured dictionary, the algorithm does not require a diverse or a large dataset for training and is computationally more efficient than alternative methods. The performance of the proposed structured dictionary transcription system is empirically examined, and its advantage is demonstrated compared to alternative dictionary learning methods.

© 2012 Elsevier Inc. All rights reserved.

1. Introduction

Transcription of music is defined as the process of identifying the parameters of an acoustic musical signal, which are required in order to write down the score sheet of the notes [1]. One of the most important parameters is the pitch, which is represented in written music by the note symbol. For convenience, we shall refer here to the task of pitch identification as “transcription”. Automatic transcription of music is important, since it allows structured audio coding, it is a helpful tool for musicians, enables modifying, rearranging and processing music in an efficient way, and is the basis for interactive music systems.

We need to separate between two different cases – transcription of monophonic music and transcription of polyphonic music. Monophonic music is the case in which a single note is played at each time instant. For this case, automatic transcription is practically a solved problem. Several proposed algorithms are reliable, commercially applicable and operate in real time. However, transcription of polyphonic music, in which more than one note is played at a time, is much more complicated. To the best of our knowledge, no existing algorithm can identify multiple pitches in an accuracy close to 100%. This is somehow not intuitive to understand, since when a trained human (such as a musician) listens to a polyphonic music piece, he can distinguish between the different notes and identify them, although played simultaneously. The difficulty stems from the fact that most often, especially in western music, several harmonics of different notes overlap. This causes a problem in assigning the harmonics to their true fundamental frequencies, and in deducing spectra of several notes from their sum [2,3].

Since the 1970s, when Moorer built a system for transcribing duets [4], there has been a growing interest in transcribing polyphonic music. The offered algorithms can be divided into three main groups: time-based, frequency-based and time–frequency-based algorithms. The time-based group includes methods which are based on the autocorrelation function [5–7] and on the periodicity of the signal [8–10]. The frequency-based group includes methods which are based on typical harmonic patterns in the frequency domain [11,12], and which can be mapped to a logarithmic scale to better fit the human auditory system [13–16]. The combined time–frequency-based group includes methods which use a time–frequency image, such as a spectrogram or a scalogram [17], or cepstrum analysis [18,19].

Some of the above methods [11,19] are based on the assumption that the spectral shape of the harmonics can be modeled by a constant function, therefore spectrums of several notes can be deduced from their combination. This is very inaccurate, since the spectral shape changes as a function of many factors, which include the type of the musical instrument, the total intensity of the
tone, the fundamental frequency and the stage in the time enve-
do the ADSR (attack, decay, sustain, release) envelope of the note [20]. Other methods [10,12,15,21] are limited by the fact that they are supervised methods, i.e., require a training set of pieces in order to transcribe another musical piece.

The idea of using Sparse representations as a time-based or frequency-based method for transcription of polyphonic music, was first suggested by Abdallah and Plumbley [22]. It was later improved and expanded [8,23,24], and inspired other works [25–27].

The term “Sparse representations” means writing a signal as a linear combination of very few underlying functions, which are contained in a dictionary of underlying functions. This is implemented by multiplying the dictionary of the underlying functions by a sparse vector (i.e., a vector that contains very few non-negative elements compared to its length), giving the method its name.

The motivation for applying sparse representations of music signals, is that in played music, only a small number of notes is played simultaneously compared to the number of available notes. In the frequency domain, the approach is based on the idea that power spectra of different notes approximately add, assuming random phase relationships. However, the existing algorithm for transcription of music using sparse representations, developed by Abdallah and Plumbley [24], suffers from a major drawback – it requires a large and diverse database of notes, in order to build a representative and reliable dictionary for the transcription. For example, if a note is played only in combination with other notes, the learned dictionary will not represent the individual note, but the combination in which it resides.

Several researchers, such as Raczynski et al. [28], Hennequin et al. [29], Ewert and Müller [30] and Vincent et al. [31–33], used Non-negative Matrix Factorization (NMF) for transcription of polyphonic music. Some of their work involved representing the signal using an adaptive structured decomposition, which is quite similar to our approach. However, while they used NMF for the decomposition, we used sparse representation and structured dictionary learning, which involve different updating of the decomposed parameters. A more detailed comparison is introduced in the Conclusions.

In this paper, we present an algorithm for multiple pitch estimation based on sparse representations in a structured dictionary, suitable for the spectra of music signals. It achieves good estimation results even with a small and limited database of notes, thus overcoming the drawback of the former algorithm based on sparse representations. We show the advantages of our algorithm over other algorithms for transcription of polyphonic music, using experiments on synthesized and recorded piano music.

The structure of the paper is as follows: In Section 2, we describe the musically-structured algorithm, based on sparse representations and spectral analysis of music signals. In Section 3, we review the implementation of the algorithm in a transcription setup for polyphonic music. In Section 4, experimental results are presented, and the algorithm is compared to other transcription methods. In Section 5, we conclude the paper and suggest future expansions.

2. Musically-Structured (MS) sparse representations

Our transcription algorithm is based on sparse representations with our new parametric dictionary, suitable for the spectra of music signals. We define this dictionary as the Musically-Structured dictionary. The algorithm is composed of two stages: Sparse coding and structured dictionary learning.

2.1. Sparse coding

The purpose of sparse coding is to approximate the solution of the following (P_0) problem:

\[
(P_0): \min_{x} \|x\|_0 \quad \text{subject to} \quad Ax = y. \tag{1}
\]

The vector x is a sparse vector, encoding the columns (atoms) of the dictionary A, such that together they represent the signal y. This method was shown useful to capture the meaningful characteristics of signals, such as images and audio [34].

An error-tolerant modified version for multiple signals:

\[
(P_0^P): \min_{X} \|Y - AX\|_F^2 \quad \text{s.t.} \quad \|x_i\|_0 < K, \ 1 \leq i \leq M, \tag{2}
\]

where F is the Frobenius norm, and x_i are the columns of X. In our case, each of the columns of the matrix Y, i.e., the signals, is the Constant Q Transform (CQT) [35], at a certain time window. We use the CQT instead of the Short-Time Fourier Transform (STFT), since it is more suitable for the auditory system. The STFT can be modeled as a bank of linearly spaced filters, whose bandwidths are constant. On the other hand, the CQT can be modeled as a bank of logarithmically spaced filters, whose bandwidths are logarithmically increasing as a function of frequency. The Q parameter, which is the bandwidth to center-frequency ratio, is constant as a function of the frequency. In the original CQT transform, the length of the time window decreases as a function of frequency. We use a degenerated version of it, with a constant window length. Because of its logarithmic frequency scale, the CQT better fits the auditory system than the STFT [35], and also reduces the computational complexity by using less frequency bins than the STFT.

The matrix A ∈ R^{m×n} is the Musically-Structured dictionary in which each atom (column) is the CQT of a single note. Each column in X ∈ R^{M×M} encodes a linear combination of the notes from A which are played in the time window of the corresponding column in Y ∈ R^{N×M}. At each time window, the corresponding column in X is determined as the sparsest column which minimizes the Frobenius norm \|Y - AX\|_F. In our algorithm we use a greedy method for sparse coding, since it allows to pre-define the cardinality (the number of the non-zero elements K) of the sparse vector, according to the evaluated number of notes at each time window. Specifically, we choose to use the OMP algorithm [36,37], which settles a good compromise between complexity and performance, compared to other algorithms for sparse coding, such as Matching Pursuit (MP) or Basis Pursuit (BP) algorithms [34].

2.2. Musically-Structured dictionary

The dictionary matrix A can be chosen in three different manners:

1. (1) Analytic dictionary – a predefined dictionary, which is the inverse (or pseudo-inverse) matrix of a transform such as the Curvelet transform [38,39], the Contourlet transform [40,41] the short-time Fourier transform or the Wavelets transform [42].

2. Learned (explicit) dictionary – a dictionary is learned blindly from a set of signals or samples y_{i,1≤i≤M}, in an iterative and alternative manner, arranged as columns of a matrix Y. The optimization problem (P_0^P) turns into:

\[
\min_{A,X} \|Y - AX\|_F^2 \quad \text{s.t.} \quad \|x_i\|_0 < K, \ 1 \leq i \leq M. \tag{3}
\]

Abdallah and Plumbley [22,24] used such a learned dictionary for transcription of polyphonic music.
(3) Parametric dictionary – a compromise between a pre-defined analytic dictionary to a data-driven learned dictionary. Only a limited set of parameters in the dictionary is learned, and the rest is pre-defined. It represents the data better than an analytic dictionary, and is computationally more efficient than an explicit learned dictionary. In addition, it can avoid overfitting if built wisely, unlike an explicit learned dictionary.

Our motivation is to develop a parametric dictionary which is suitable for the spectrum of music signals. In order to do so, we use the common features in the spectrum of a musical note. The magnitude of the Fourier transform of every note can be modeled by an impulse train at the fundamental frequency and at its harmonics, which has been multiplied with a shaping filter, that is controlled by the factors mentioned in Section 1 – the musical instrument, the intensity, the duration of the note and the fundamental frequency. For illustration, see the short-time Fourier transform (STFT) of an A4 piano note in Fig. 1. This STFT illustration was performed on a note played over a long time, so we could use a large time window, and get good resolution both in time and in frequency.

If we apply the Constant Q transform instead of the STFT on a A4 piano note, we get peaks of the fundamental frequency and its harmonics in intervals which become logarithmically smaller as the CQT bin grows (see bottom of Fig. 2). The dictionary which we offer is initialized such that each atom is the modeled CQT of a different note in the piano (total – 88 notes). More specifically, we initialize each atom by an impulse train of 6 elements, corresponding to the suitable fundamental frequency and its first 5 harmonics. This number of harmonics was found to be sufficient in multiple tests and on a wide range of notes. We multiply this finite impulse train by an initial shaping function $f(n) = \frac{1}{n}$, where $n$ is the partial number. Finally, we map the dictionary to the CQT scale. We denote this dictionary as the initial Musically-Structured dictionary. An image of this initial dictionary, and the CQT of a note represented by a certain atom in it, are presented in Fig. 2.

The next step is learning certain parameters in the Musically-Structured dictionary. The support of the dictionary is constant, i.e., the location of the non-zero elements in the vector, which are the CQT bins of the fundamental frequencies and their 5 first harmonics. However, the entries of the elements in the support are learned, i.e., the amplitudes of the fundamental frequencies and their harmonics. These amplitudes are an important feature of the timbre of the signal, which was mentioned before. The minimal resolution required for music transcription is 12 CQT frequency bins per octave (one for each 100 cent = semitone). However, using a higher resolution improves the transcription results for polyphonic music. In the experiments presented in Section 4 we show the results for a mapping to 24 frequency bins per octave (one for each 50 cents = quarter tone).

The MS-dictionary is expected to have several advantages over the explicit learned dictionary proposed by Abdallah and Plumbley [24], as well as advantages over an analytic and pre-defined appropriately dictionary:

(1) Avoids overfitting – each atom in the MS dictionary represents a single note, even if it does not appear individually in the dataset (e.g., it is played only as part of a chord). We expect this requirement to be fulfilled, since the support of the dictionary is constant, and since each atom in the initial dictionary represents an individual note.

(2) Better representation of the signal than an analytic dictionary – this is due to the fact that the entries in the support of the dictionary matrix are learned according to the timbre of the signal.

(3) Reduced complexity – the complexity of learning the dictionary is proportional to the number of the learned parameters. In an explicit dictionary, the number of learned parameters equals the size of the dictionary, i.e., $nm$. In the MS-dictionary,
the number of learned parameters is \(hm\), where \(h\) is the number of partials (fundamental frequency + harmonics), which we choose to be 6. Since \(h < n\), the complexity of the dictionary is reduced.

The formulation of the new problem is:

\[
(P_{MS}): \quad \min_{AX} \|Y - AX\|^2_F \quad \text{s.t.} \quad \|x_i\|_0 \leq K \forall i \in [1, \ldots, m]
\]

where \(P_j^k\) is the subset of indices out of the support of the atom \(a_j\).

In order to approximate the solution to \(P_{MS}\), we apply a modified MOD (Method-Of-Directions) or K-SVD dictionary learning algorithm according to the MS parametric dictionary.

The MS-dictionary is composed of 88 atoms, where each atom represents a different note. Musical pieces usually do not contain all of the 88 notes. Therefore, in the dictionary learning stage, we update only the atoms which were used for the sparse coding stage. We denote the dictionary matrix of the used atoms in the \(k\)th iteration as \(\tilde{A}_{(k)}\), and the corresponding coefficients matrix as \(\tilde{X}_{(k)}\). The rest of the atoms remain unchanged, and are added to the dictionary after the update. In the following, we describe the modified MOD and K-SVD dictionary learning algorithms.

### 2.3. Musically-Structured MOD

In the MOD algorithm, developed by Engan et al. [43], the dictionary update stage is conducted using least squares:

\[
A_{(k)} = \arg\min_{A} \|Y - AX_{(k)}\|^2_F = YX_i^T,
\]

where \(X_i^T\) is the Moore–Penrose pseudo-inverse of \(X\). After an initialization of the dictionary matrix, the matrix of sparse columns \(X\) and the dictionary matrix \(A\) are updated alternately at each iteration, until the change at the \(k_{th}\) iteration of \(\|Y - A_{(k)}X_{(k)}\|^2_F\) is small enough.

In our modified MOD algorithm, the MS-MOD, the dictionary matrix is updated as in the MOD algorithm:

\[
\tilde{A}_{(k)} = \arg\min_{A} \|\tilde{Y} - \tilde{A}X_{(k)}\|^2_F = \tilde{Y}X_i^T.
\]

The matrix \(\tilde{X}\) is composed of the concatenation of the rows of \(X\) which have a \(\ell_1\)-norm above a certain threshold, and correspond to the used atoms in \(A\). This diminution is intended to prevent singularity of \(X_i\), and deals with most of the cases, where the musical piece does not contain all of the 88 notes.

The normalization stage of the atoms in the MOD (and in the K-SVD) is intended to make the implementation of the sparse coding stage simpler when using a greedy method, and it does not change its solution.

After updating the atoms, we zero the elements out of the original support. The MS-MOD is summarized in Table 1. The main modifications compared to the original MOD are indicated by underlined text.

### 2.4. Musically-Structured K-SVD

A different update rule for the dictionary was proposed by Aharon et al. [44], leading to the K-SVD algorithm. In this algorithm, the atoms (i.e., columns) in the dictionary \(A\) are handled sequentially. The dependency on the atom \(a_{j_0}\) in (3) is isolated by rewriting the term \(\|Y - AX\|^2_F\):

\[
\|Y - AX\|^2_F = \left\| Y - \sum_{j \neq j_0} a_j x_j^T \right\|^2_F - \|a_{j_0} x_{j_0}^T\|^2_F \quad (7)
\]

In this description \(x_j^T\) stands for the \(j\)th row of \(X\), i.e., the coefficients which correspond to the \(j\)th atom. We define the term

\[
E_{j_0} = Y - \sum_{j \neq j_0} a_j x_j^T \quad (8)
\]

as the error matrix corresponding to the atom \(a_{j_0}\). We restrict it only to the columns that correspond to the signals (columns) in \(Y\) which use the atom \(a_{j_0}\), and denote the restricted error matrix as \(E_{j_0}^R\). Both \(a_{j_0}\) and the non-zero elements in \(x_{j_0}^T\) which are denoted by \((x_{j_0}^R)^T\), are updated in this algorithm, by minimizing the term in (7), using a rank-1 approximation of the error matrix \(E_{j_0}\).

This approximation is obtained via singular value decomposition (SVD).

In our modified algorithm, the MS-KSVD, we update only the elements in the support, for each atom individually. For each atom \(a_{j_0}\), the error matrix \(E_{j_0}\) is defined as in the K-SVD. Its columns are restricted as in the K-SVD, but now its rows are also restricted, according to the support of the atom \(a_{j_0}\). Thus, updating only the support of \(a_{j_0}\), we denote this restricted error matrix as \(E_{j_0}^R\) and the elements in the support of \(a_{j_0}\) as \(\tilde{a}_{j_0}\). The vectors \(\tilde{a}_{j_0}\) and \((x_{j_0}^R)^T\) are updated using rank-1 approximation of the error matrix \(E_{j_0}^R\) by singular value decomposition. The algorithm is summarized in Table 2. The main modifications compared to the original K-SVD are indicated by underlined text.

### 3. Implementation

The overall transcription algorithm which we offer is as follows:

1. Note onset detection – since we focus only on pitch identification, we conduct this stage manually, or extract the onsets

<table>
<thead>
<tr>
<th>Table 1</th>
<th>The Musically-Structured MOD algorithm.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Task:</strong></td>
<td>Train a dictionary (A) to sparsely represent the log spectrogram matrix (Y), by approximating the solution to Problem (4).</td>
</tr>
<tr>
<td><strong>Initialization:</strong></td>
<td>Initialize (k = 0), and</td>
</tr>
<tr>
<td>- Initialize Dictionary:</td>
<td>Build (A_{(0)} \in \mathbb{R}^{m \times m}) with non-zero entries only in the locations corresponding to the fundamental frequencies and their harmonics.</td>
</tr>
<tr>
<td>- CQT mapping:</td>
<td>Map the dictionary to the CQT logarithmic scale.</td>
</tr>
<tr>
<td>- Normalization:</td>
<td>Normalize the columns of (A_{(0)}) to unit (\ell_2)-norm.</td>
</tr>
<tr>
<td><strong>Main iteration:</strong></td>
<td>Increment (k) by 1 and apply</td>
</tr>
<tr>
<td>- Sparse Coding Stage:</td>
<td>Use a pursuit algorithm to approximate the solution of</td>
</tr>
<tr>
<td>- Define the used atoms:</td>
<td>The rows in (X_{(k)}) which have an (\ell_1)-norm above a certain threshold correspond to the used atoms in the dictionary matrix. These rows are concatenated and defined as (\tilde{X}<em>{(k)}), and the matrix composed of the used atoms is defined as (A</em>{(k)}).</td>
</tr>
<tr>
<td>- MOD Dictionary Update Stage:</td>
<td>Update the dictionary of used atoms (A_{(k)}), by the matrix (X_{(k)}):</td>
</tr>
<tr>
<td>- Zero the entries out of the support:</td>
<td>Zero the entries out of the support which was defined in the initial dictionary.</td>
</tr>
<tr>
<td>- Add the unused atoms:</td>
<td>Add the unused atoms to (A_{(k)}).</td>
</tr>
<tr>
<td><strong>Stopping Rule:</strong></td>
<td>If the change in (|Y - A_{(k)}X_{(k)}|^2_F) is small enough, stop. Otherwise, apply another iteration.</td>
</tr>
</tbody>
</table>

**Output:** The desired outputs are \(A_{(k)}\) and \(X_{(k)}\).
4. Experiments

4.1. Experimental setup

We perform pitch identification of synthesized piano music from MIDI files, and pitch identification of real recorded piano music. We do not deal with finding the onset and offset instances of the notes, neither with finding the number of notes played at each time (which we use as the maximal number of notes – \( K \), for the sparse coding). The stopping criterion for the algorithms is achieved when there is a change of less than 5\% in the Frobenius norm of the residual. We compare the performance of the MS-MOD and MS-K-SVD to that of the standard MOD and K-SVD algorithms (which use an explicit, blindly learned dictionary), and to that of an analytic dictionary (un-learned) with OMP in the sparse coding stage. The analytic dictionary is determined as the initial MS-dictionary. In the standard MOD and K-SVD we decide on the fundamental frequency of individual atoms that are learned, by a method called HPS (Harmonic Product Spectrum) [45] which is intended for pitch estimation in monophonic music.

We use different measures for the evaluation of the results: The first is the Accuracy measure, defined by [46].

\[
\text{Accuracy} = \frac{TP}{TP + FP + FN}.
\]

The term TP is the number of true positives (correct detections), FP is the number of false positives and FN is the number of false negatives. When Accuracy = 1, it means that all the notes are identified correctly and there are no false positives nor false negatives. The second measure is the transcription error score [12]. If we denote by \( N_{\text{sys}} \) the number of reported pitches, by \( N_{\text{ref}} \) the number of ground-truth pitches and by \( N_{\text{corr}} \) their intersection, then the transcription error score across all time frames \( t \) is:

\[
E_{\text{tot}} = \frac{\sum_{t=1}^{T} \max(N_{\text{ref}}(t), N_{\text{sys}}(t)) - N_{\text{corr}}(t)}{\sum_{t=1}^{T} N_{\text{ref}}(t)}.
\]

The MIDI files for the experiments include a monophonic musical piece, a simple polyphonic piece (mainly two or three notes played at a time, a limited variety of notes), a complicated polyphonic piece (mainly five or six notes played at a time, a large variety of notes) and a piece of chords. Their piano rolls are presented in the next subsection. These MIDI files are synthesized with a sampling frequency of 44.1 kHz, by FM-synthesis, using the “Matlab and MIDI” software [47]. The MS-K-SVD code is a modified version of the K-SVD code [48]. The recorded piano files include a monophonic piece and a piece of chords. These pieces are recorded on a Yamaha U1 piano, and saved with a sampling frequency of 44.1 kHz.

We also compare our transcription method to former reported transcription results (Costantini et al. [15], Poliner and Ellis [12], Rynänen and Klapuri [21] and Marolt [10]), which were examined on a set of polyphonic classical synthesized MIDI music pieces which were collected from the Classical Piano Midi Page [49]. The list of 130 pieces set is specified in [12]. The first minute from each song was taken. The 130 pieces set was randomly split into 92 training, 24 testing and 13 validation pieces (we used for our tests only the testing set since no training is needed). In addition to the

4.2. Results

The results are presented in Table 1. The first section presents the results for the synthetic piano music, while the second section presents the results for the recorded piano music.

Table 1

<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy</th>
<th>Error Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOD</td>
<td>0.98</td>
<td>1.2 %</td>
</tr>
<tr>
<td>K-SVD</td>
<td>0.96</td>
<td>1.5 %</td>
</tr>
<tr>
<td>Analytic</td>
<td>0.94</td>
<td>1.8 %</td>
</tr>
<tr>
<td>OMP</td>
<td>0.92</td>
<td>2.0 %</td>
</tr>
</tbody>
</table>

The results show that the MOD algorithm achieves the highest accuracy, while the OMP method performs the worst. The error score is also the lowest for the MOD method, indicating better performance.

Fig. 3. A block diagram of the overall transcription algorithm.
synthesized audio, piano recordings were made from a subset of the classical MIDI files using a Yamaha Disklavier playback grand piano. Twenty training files and ten testing files were randomly selected for recording. The recorded files are available at [50].

4.2. Results – synthesized MIDI music

First we present the performance of the MS-dictionary learning algorithm on a monophonic piece, a simple polyphonic piece and a complicated polyphonic piece. The transcription results are presented in Tables 3 and 4. The corresponding piano rolls and their identification by MS-MOD and MS-KSVD are presented in Figs. 4, 5 and 6. From the tables we can learn that:

1. All methods perfectly identify the notes in the monophonic music.

<table>
<thead>
<tr>
<th>Table 3</th>
<th>Transcription Accuracy percentage for three different types of songs.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Monophonic music</td>
</tr>
<tr>
<td>MS-MOD</td>
<td>100</td>
</tr>
<tr>
<td>MS-K-SVD</td>
<td>100</td>
</tr>
<tr>
<td>MOD</td>
<td>100</td>
</tr>
<tr>
<td>K-SVD</td>
<td>100</td>
</tr>
<tr>
<td>Analytic dictionary</td>
<td>100</td>
</tr>
</tbody>
</table>

2. The estimation performance of the algorithms based on the parametric MS dictionary is better than that of the explicit dictionaries as well as the analytic dictionaries when identifying notes in polyphonic music. The advantage is more significant in the simple polyphonic music than in the complicated polyphonic music (where the dataset of notes is larger and richer).

3. The estimation performance of the MS-dictionary and the analytic dictionary become worse when the polyphonic music becomes more complicated. On the contrary, the performances of the methods based on explicit dictionaries are improved. This implies that the explicit dictionaries need a large data-set in order to achieve good performance.

From the piano rolls we can identify some mistakes in the transcription as spurious notes (mistakes of semitones), which might be fixed by using a higher frequency resolution. Some mistakes are associated with notes that share a similar spectral shape, such as notes whose difference is an integer number of octaves, or notes that share common harmonics with the true note.

We now turn to the problem mentioned in Section 1 – deduction of notes from chords. This problem is hard, since the notes in a chord share multiple harmonics. We compare the results of the MS-dictionary to that of an explicit dictionary, and of an analytic dictionary. The piano rolls of the original and identified music are presented in Fig. 7. In the case of the chords, MS-MOD, MS-K-SVD and OMP with an analytic dictionary identify all the notes, despite of the difficulty of this task. The MOD and K-SVD identify only the lower notes.

We compare the former reported results, obtained by the implementations of original authors (Costantini et al. [15], Poliner and Ellis [12], Ryynänen and Klapuri [21] and Marolt [10]), on the set of polyphonic classical music, which was described in the beginning of this section, to the multi-pitch estimation of the testing set by MS-MOD, MS-K-SVD, MOD, K-SVD and OMP with an ana-

Fig. 4. The ground-truth piano roll (left) and the identified piano roll of a monophonic piece using MS-MOD (middle) and using MS-K-SVD (right). All the notes were identified correctly.

Fig. 5. The ground-truth piano roll (left) and the identified piano roll of a simple polyphonic piece using MS-MOD (middle) and using MS-K-SVD (right). Black = true positive, red = false positive, yellow = false negative. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)
lytic dictionary. However, the results should be compared carefully due to the following differences between the algorithms:

1. The former transcription algorithms are supervised methods, i.e. they are based on the training set mentioned before. Their results presented here are after training on pieces which were written by the same composers as in the testing set. Our algorithm, and the other transcription methods based on sparse representations (MS-MOD, MS-K-SVD, MOD, K-SVD and OMP with an analytic dictionary), as the maximal cardinality $K$ of each vector of coefficients $x$. However, the former transcription algorithms did not receive the maximal cardinality as a parameter.

A comparison of the results on synthesized polyphonic classical music are presented in Table 5. From the table one can see that the results of the transcription by MS-dictionary learning algorithms outperform those of the other unsupervised methods for transcription using sparse representations (MOD, K-SVD and OMP with an analytic dictionary). They are inferior compared to those of other transcription methods, but, as mentioned before, not directly comparable. The measures Accuracy and $E_{tot}$ are not always correlated in their performance, due to their different definitions. Therefore, they are both used for evaluation.

Table 5

<table>
<thead>
<tr>
<th></th>
<th>Accuracy (%)</th>
<th>$E_{tot}$ (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MS-MOD</td>
<td>58.7</td>
<td>28.2</td>
</tr>
<tr>
<td>MS-K-SVD</td>
<td>59.8</td>
<td>27.2</td>
</tr>
<tr>
<td>MOD</td>
<td>39.4</td>
<td>45.3</td>
</tr>
<tr>
<td>K-SVD</td>
<td>31.0</td>
<td>54.1</td>
</tr>
<tr>
<td>Analytic dictionary</td>
<td>39.1</td>
<td>45.1</td>
</tr>
<tr>
<td>Costantini et al.</td>
<td>72.3</td>
<td>20.1</td>
</tr>
<tr>
<td>Poliner and Ellis</td>
<td>64.7</td>
<td>41.7</td>
</tr>
</tbody>
</table>
4.3. Results – recorded piano music

The task of multiple pitch estimation of real recorded piano music is much harder, since it entails several additional obstacles, such as beginning transients of notes, a noisy environment, formants produced by the shape of the musical instrument, reverberations of the room and strings inharmonicity. In a real vibrating string, the harmonics are not located exactly in multiple integers of the fundamental frequency, and the musical instruments are tuned accordingly [51].

First we perform some simple tests, on monophonic music and on chords. We compare the performances of MS-MOD and MS-K-SVD to that of MOD, K-SVD, and OMP with an analytic dictionary, as presented in Figs. 8 and 9. All the methods perfectly iden-
The advantage over an explicit dictionary grows as the dataset is smaller, and as there are more overlapping harmonics.

The proposed algorithm is advantageous from several reasons: It adapts to the timbre of the signal and reduces computational complexity compared to sparse representations with analytic dictionaries. It avoids overfitting to small datasets or to notes played conjugately (such as chords), thus outperforming sparse representations with explicit dictionaries. In addition, it is as unsupervised method, thus a training set is not required. Our framework can be modified and expanded in several ways: The dictionary can be better adapted to real recorded music. The overtones of real strings are not exactly multiple integers of the fundamental frequency (string inharmonicity). Therefore, a dictionary in which the non-zero elements are non-uniformly spaced, might lead to better transcription results. Another possible expansion is to add more atoms to the dictionary. The timbre of the sound changes as a function of its intensity. Therefore, it is reasonable that each note would be represented by several atoms, where each atom represents a different intensity. Each group of atoms which represents the same note would have the same support, but the values of the non-zero elements would be changed individually, such that each atom would fit another time window (or group of time windows) within the note.

Another possibility is to exploit high-level information. Former works [9,12] used prior information on the structure, tempo, rhythm and key of the musical piece, as well as expected relations between consecutive notes. This can be added to our framework as a subsequent stage, or as an iterative stage, for example via a Hidded Markov Model (HMM).

As for the comparison to non-negative matrix factorization (NMF) mentioned in Section 1, we would like to point out the similarities and dissimilarities to our work. Both approaches – NMF and sparse representations, deal with the factorization problem $\hat{A} = Y$. Moreover, sparse representations can be considered as a special case of NMF, in which the sparseness constraint is imposed [34]. In addition, in the specific NMF algorithms mentioned in Section 1, the initialization and update of the matrix $A$ are based on a special “musically-based” structure, similar to ours. However, the major difference between the NMF approach and our approach is in the update rule – while the NMF algorithms update both matrices $A$ and $X$ by minimizing the Bregman divergence between the matrix $Y$ and its approximation $AX$, we use pursuit algorithms for estimating the matrix $X$, and dictionary learning algorithms for estimating the matrix $A$. These algorithms are addressed specifically for approximating the sparse representation of the signal, with our adaptation to the problem of musical multi-pitch estimation. An interesting test would be to compare the multi-pitch estimation of our algorithm to that of the relevant NMF algorithms, on the same data set.

A next step would be to extend the proposed framework to multi-pitch estimation for other musical instruments, as well as for their combinations. It would be interesting to examine the difference in the performance of the algorithm between western and non-western instruments (e.g., [52]).

Acknowledgments

The authors thank Dr. Ron Rubinstein for fruitful discussion and helpful suggestions. They also thank the anonymous reviewers for their constructive comments and useful suggestions.

References

Dr. Cohen is a recipient of the Alexander Goldberg Prize for Excellence in Research, and the Muriel and David Jacknow award for Excellence in Teaching. He served as Associate Editor of the IEEE Transactions on Audio, Speech, and Language Processing and IEEE Signal Processing Letters, and as Guest Editor of a special issue of the EURASIP Journal on Advances in Signal Processing on Advances in Multimicrophone Speech Processing and a special issue of the Elsevier Speech Communication Journal on Speech Enhancement.